Algorithms for Bioinformatics Autumn 2011

VELI MÄKINEN

HTTP://WWW.CS.HELSINKI.FI/EN/COURSES/582670/2011/S/K/1

Lecture 6

PART I: DISTANCE-BASED CLUSTERING, UPGMA

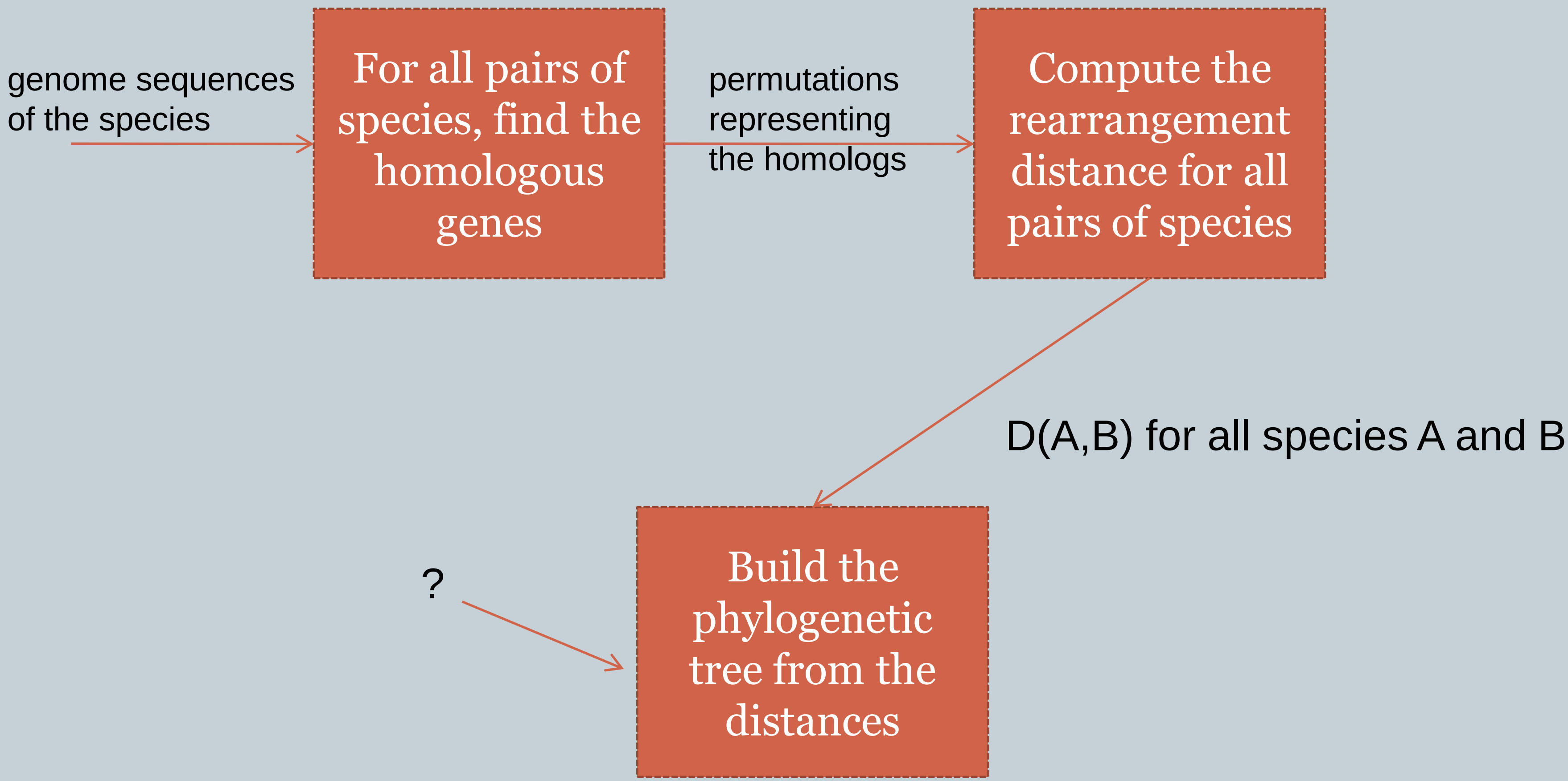
PART II: NEIGHBOR JOINING

Part of the slides are from http://bix.ucsd.edu/bioalgorithms/slides.php

Part I

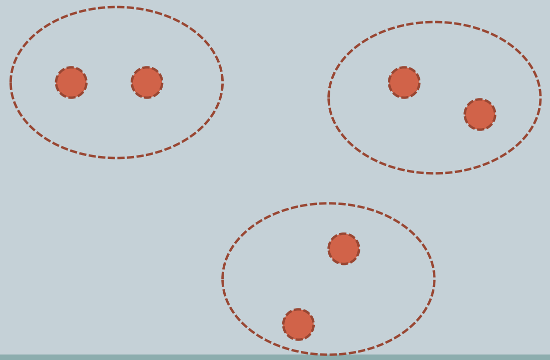
DISTANCE-BASED CLUSTERING, UPGMA

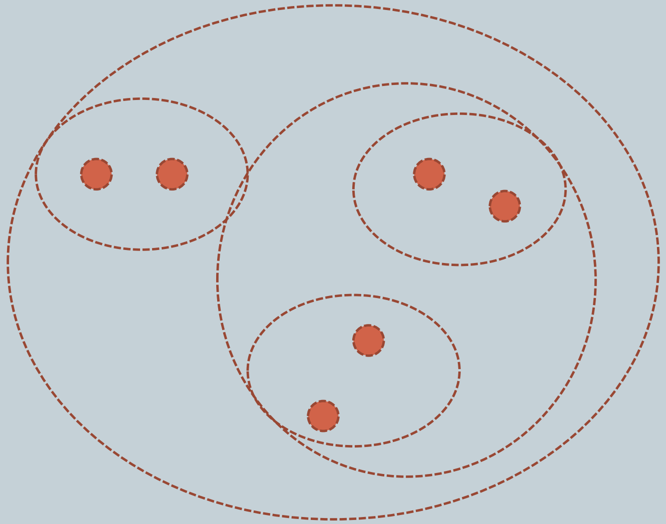
Phylogeny by distance method pipeline



Clustering

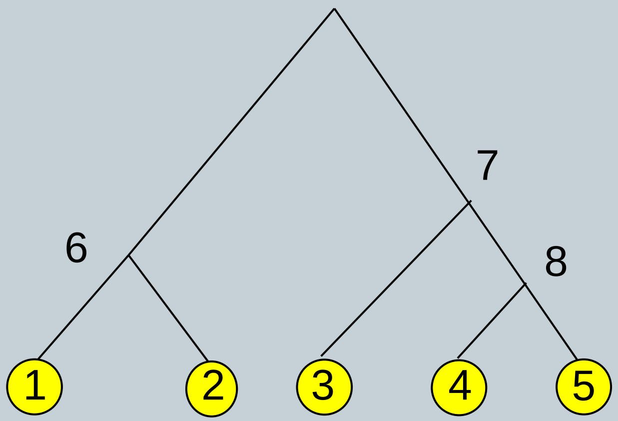
- Hierarchical clustering
  - Iteratively join two closest clusters until forming a tree hierarchy (agglomerative… also divisive version exists)
  - Distance between clusters can be e.g. max pair-wise distance (complete linkage), min (single-linkage), UPGMA (average linkage), neigbor joining
- Partitional clustering
  - k-means, etc.





Distances in a phylogenetic tree

- Distance matrix $D = (d_{ij})$ gives pairwise distances for *leaves* of the phylogenetic tree
- In addition, the phylogenetic tree will now specify distances between leaves and internal nodes
  - Denote these with d_{ij} as well



Distance d_{ij} states how far apart species i and j are evolutionary

Distances in evolutionary context

- Distances d_{ij} in evolutionary context satisfy the following conditions
  - Positivity: $d_{ij} \geq 0$
  - Identity: $d_{ij} = 0$ if and only if $i = j$
  - Symmetry: $d_{ij} = d_{ji}$ for each i, j
  - Triangle inequality: $d_{ij} \leq d_{ik} + d_{kj}$ for each i, j, k
- Distances satisfying these conditions are called *metric*
- In addition, evolutionary mechanisms may impose additional constraints on the distances
  - ▹ *additive* and *ultrametric* distances

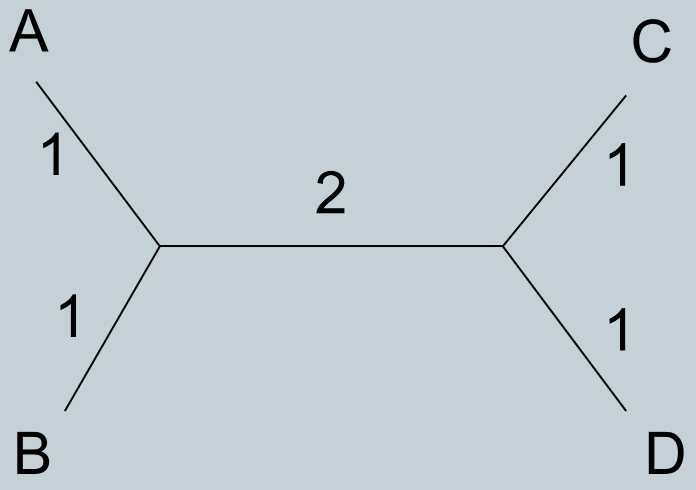
Additive trees

- A tree is called *additive*, if the distance between any pair of leaves (i, j) is the sum of the distances between the leaves and a node k on the shortest path from i to j in the tree

$$d_{ij} = d_{ik} + d_{jk}$$

Additive trees: example

	A	B	C	D
A	0	2	4	4
B	2	0	4	4
C	4	4	0	2
D	4	4	2	0



Ultrametric trees

- A rooted additive tree is called an *ultrametric tree*, if the distances between any two leaves i and j, and their common ancestor k are equal

$$d_{ik} = d_{jk}$$

- Edge length d_{ij} corresponds to the time elapsed since divergence of i and j from the common parent
- In other words, edge lengths are measured by a *molecular clock* with a constant rate

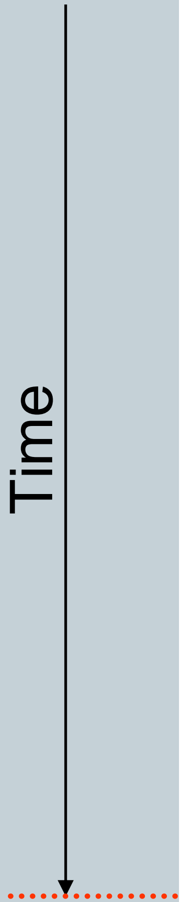
Identifying ultrametric data

- We can identify distances to be ultrametric by the three-point condition:

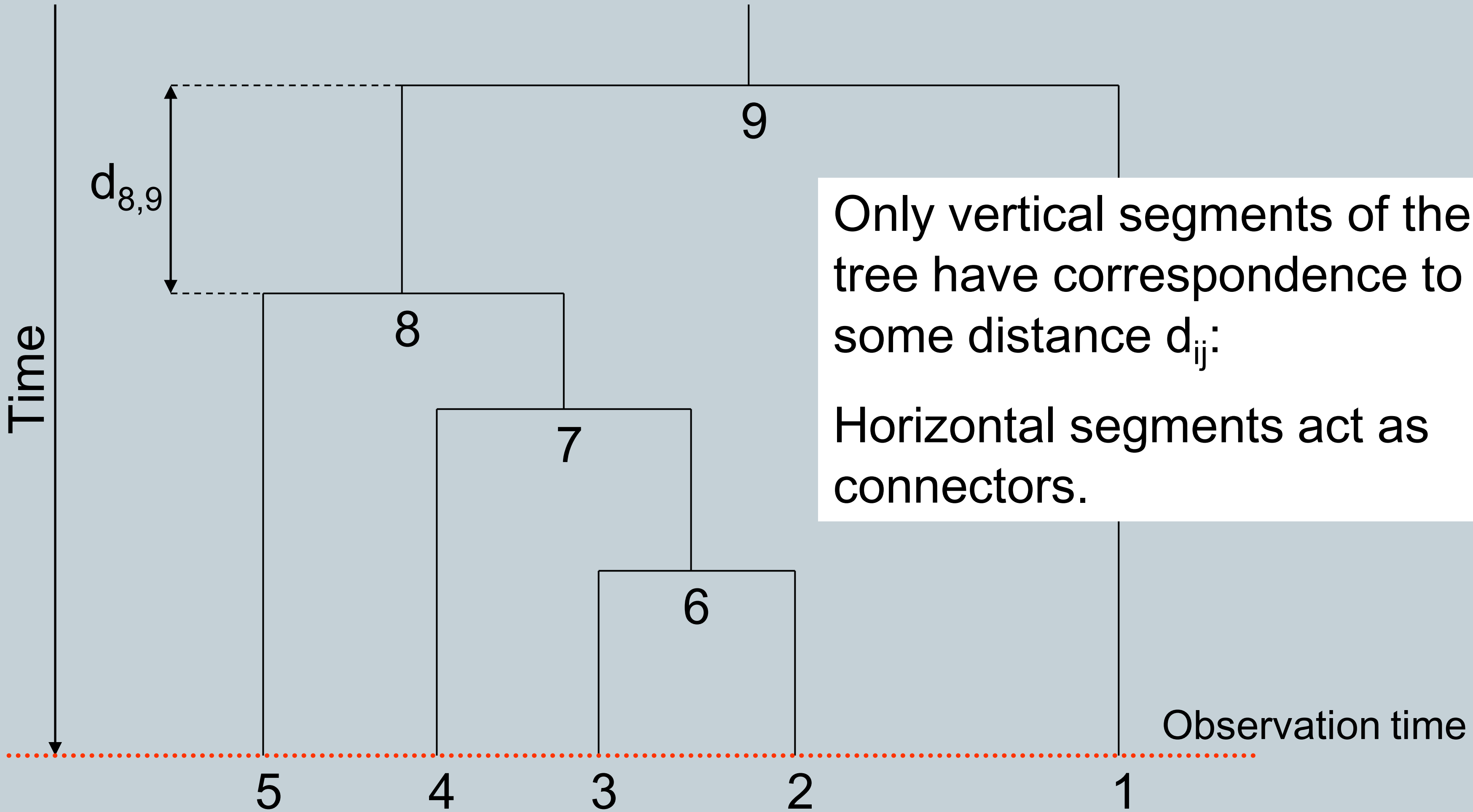
  D corresponds to an ultrametric tree if and only if for any three **species** i, j and k, the distances satisfy $d_{ij} \leq \max(d_{ik}, d_{kj})$

- If we find out that the data is ultrametric, we can utilise a simple algorithm to find the corresponding tree

Ultrametric trees



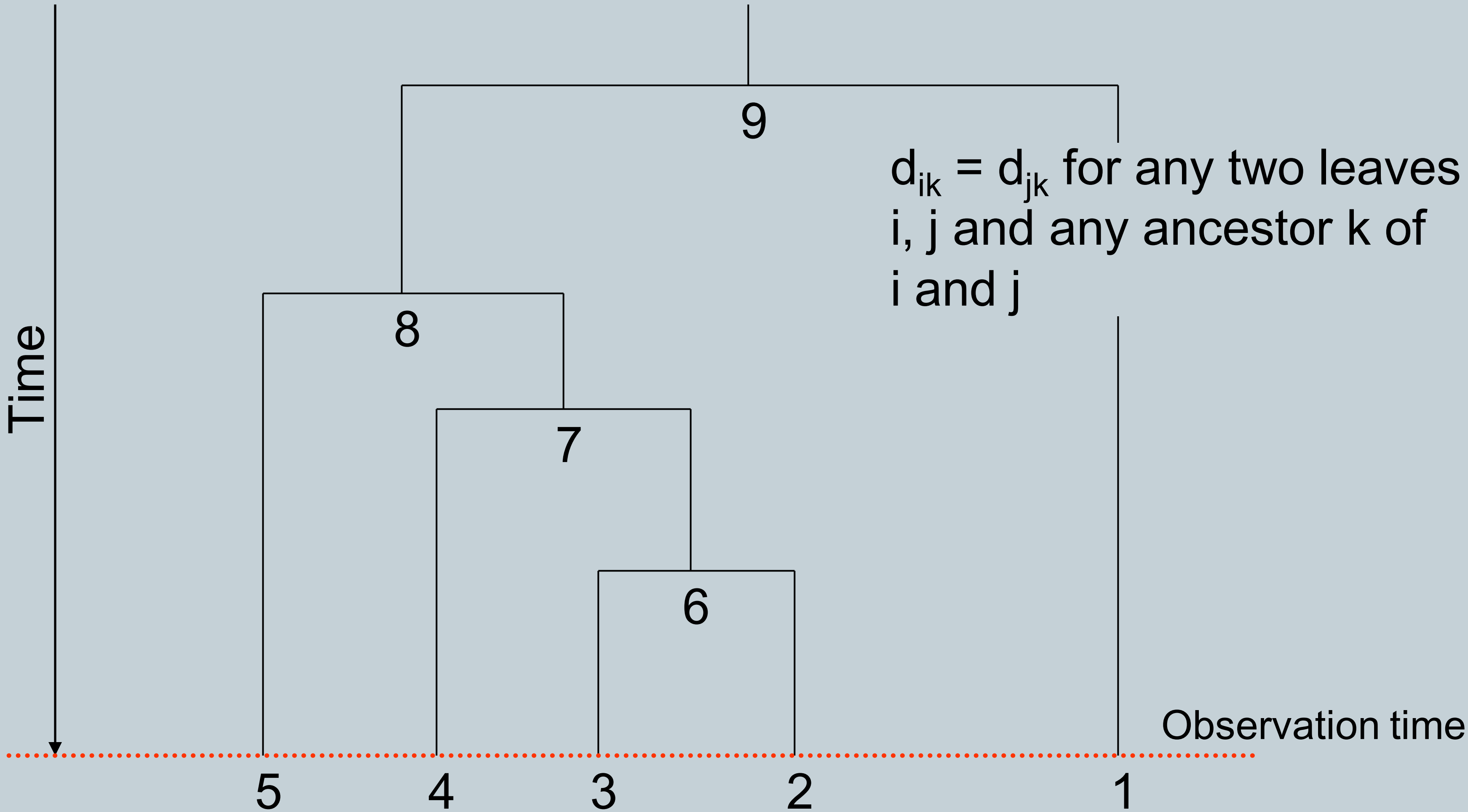
Ultrametric trees



Only vertical segments of the tree have correspondence to some distance d_{ij}:

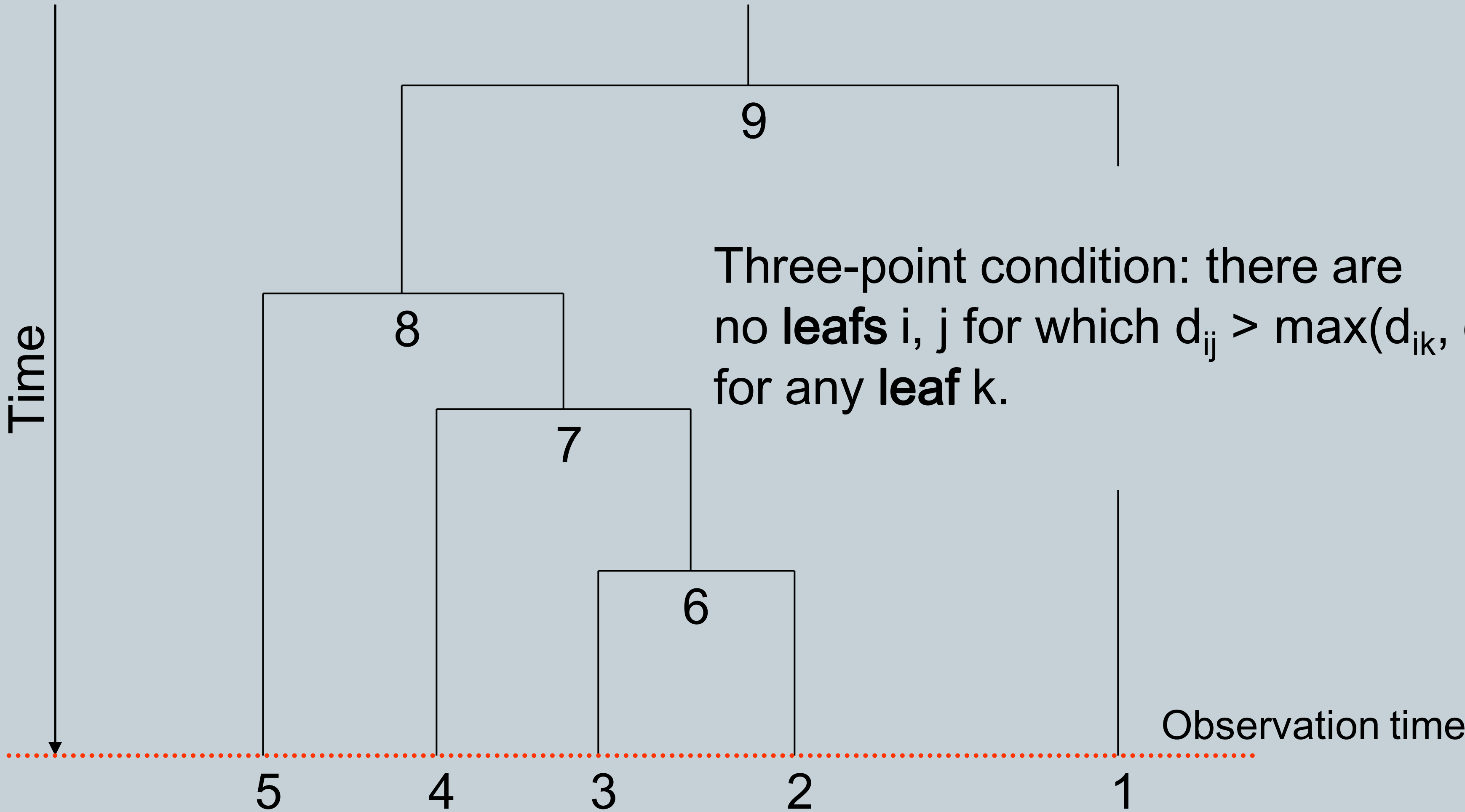
Horizontal segments act as connectors.

Ultrametric trees



$d_{ik} = d_{jk}$ for any two leaves i, j and any ancestor k of i and j

Ultrametric trees



Three-point condition: there are no **leafs** i, j for which $d_{ij} > \max(d_{ik}, d_{jk})$ for any **leaf** k.

UPGMA algorithm

- UPGMA (unweighted pair group method using arithmetic averages) constructs a phylogenetic tree via clustering
- The algorithm works by at the same time
  - Merging two clusters
  - Creating a new node on the tree
- The tree is built from leaves towards the root
- UPGMA produces a ultrametric tree

Cluster distances

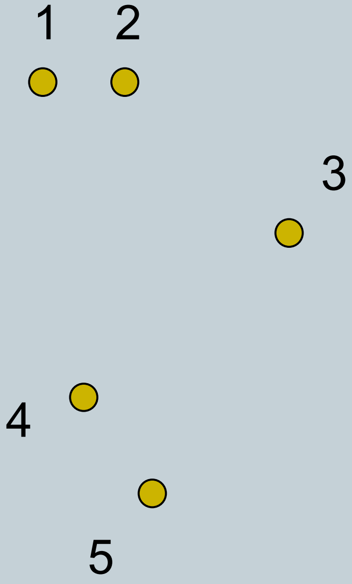
- Let distance d_{ij} between clusters C_i and C_j be

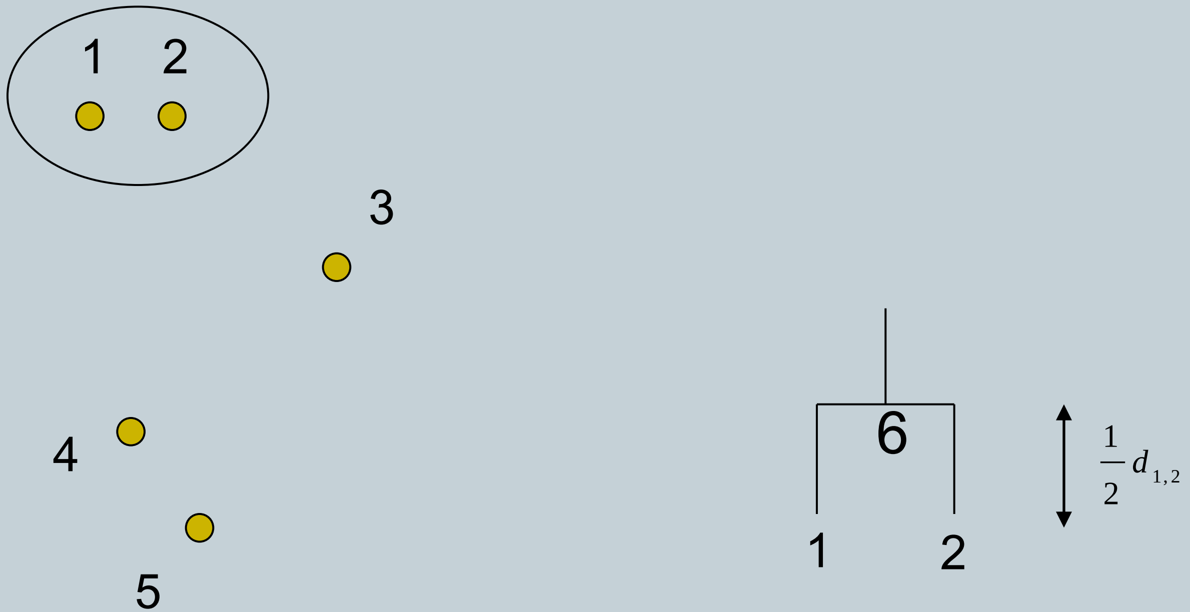
$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq},$$

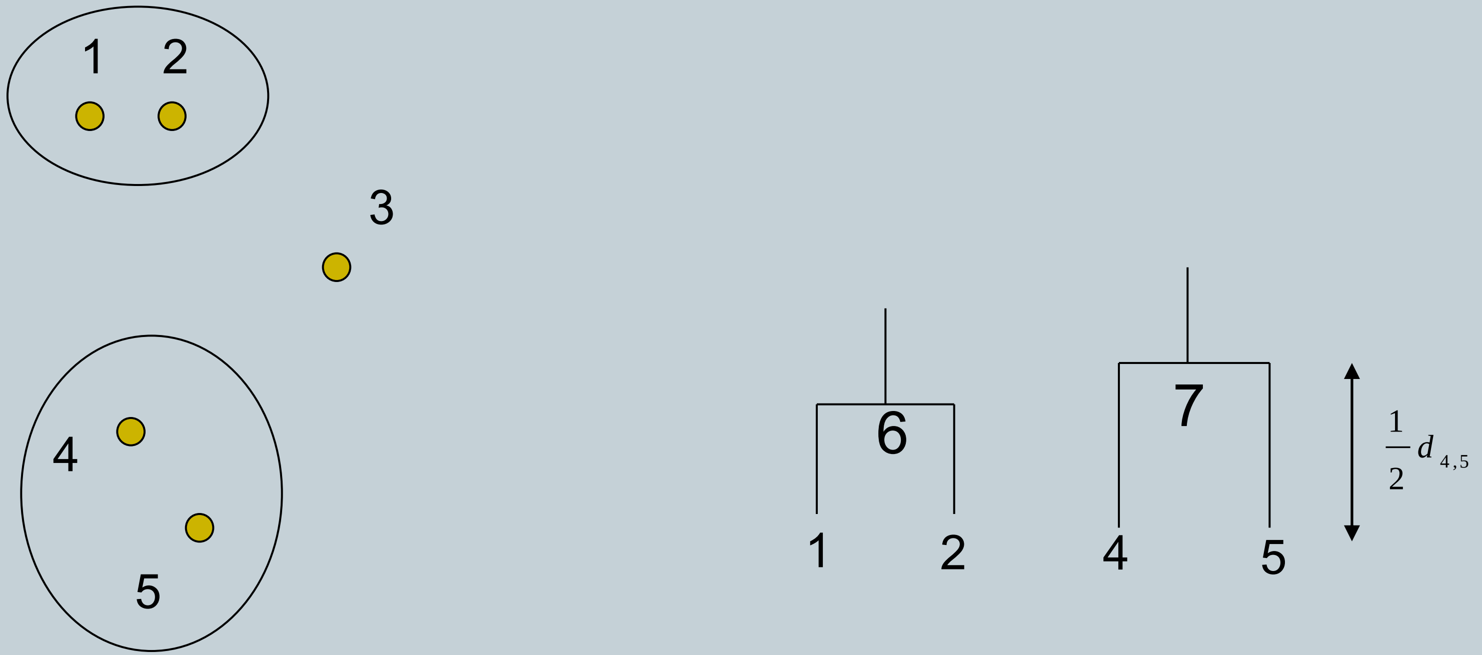
that is, the average distance between points (species) in the cluster.

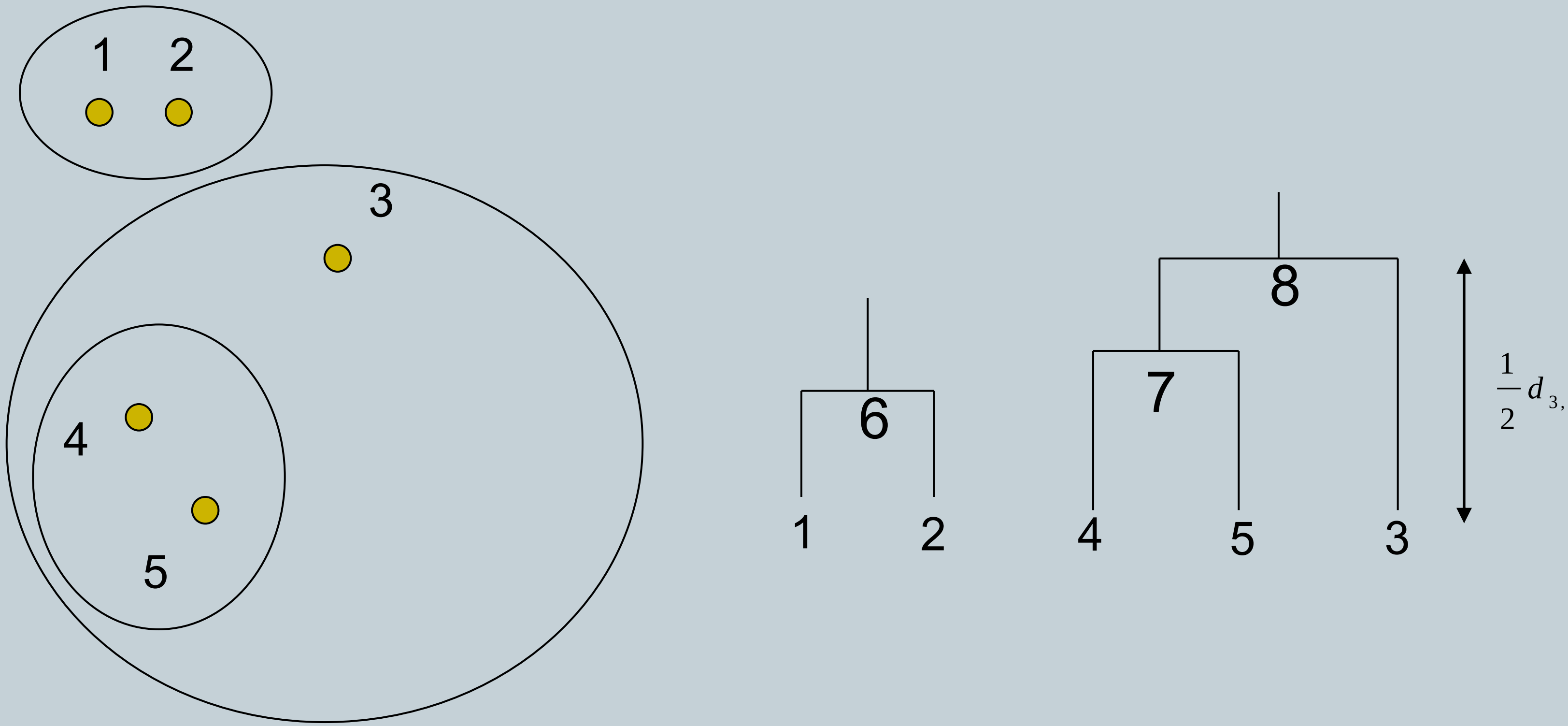
UPGMA algorithm

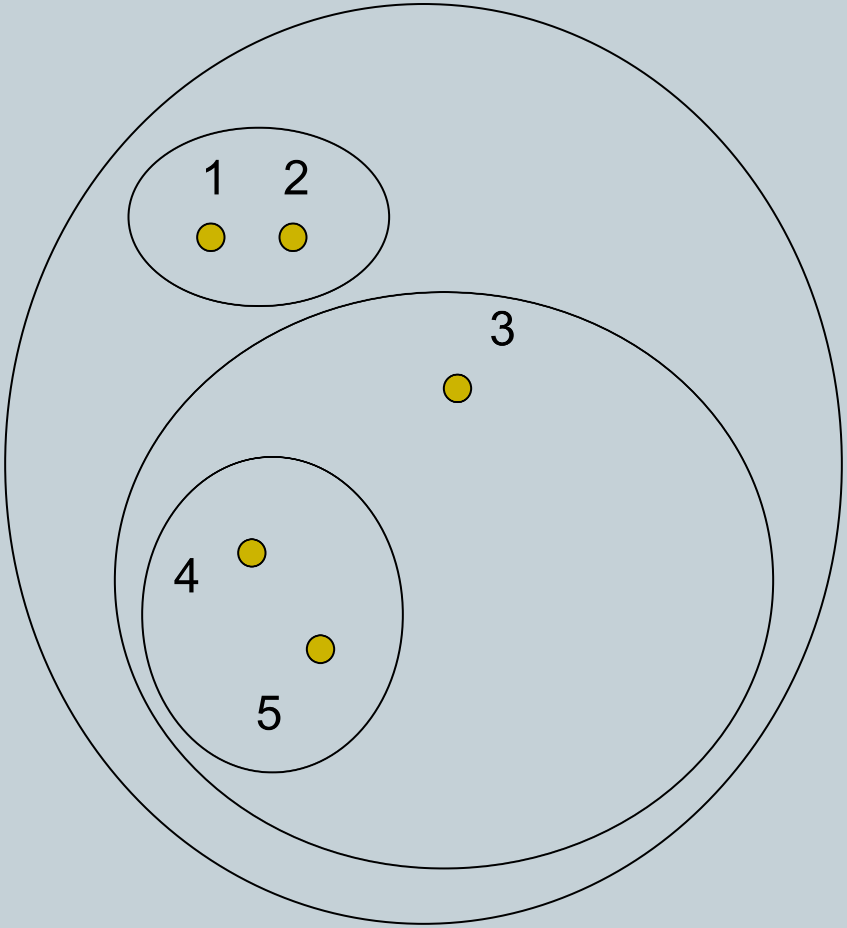
- Initialisation
  - Assign each point i to its own cluster C_i
  - Define one leaf for each sequence, and place it at height zero
- Iteration
  - Find clusters i and j for which d_{ij} is minimal
  - Define new cluster k by $C_k = C_i \cup C_j$, and define d_{kl} for all l
  - Define a node k with children i and j. Place k at height $d_{ij}/2$
  - Remove clusters i and j
- Termination:
  - When only two clusters i and j remain, place root at height $d_{ij}/2$

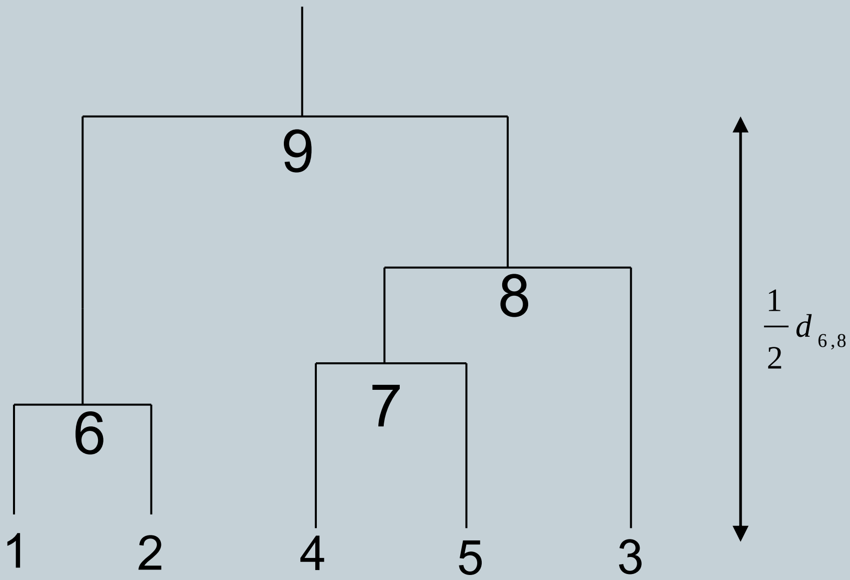

1
2
3
4
5


1
2
3
4
5
6
1
2
$\frac{1}{2} d_{1,2}$


1 2
3
4
5
6
7
1 2 4 5
$\frac{1}{2} d_{4,5}$


1
2
3
4
5
6
1
2
8
7
4
5
3
$\frac{1}{2} d_{3,7}$


1
2
3
4
5


9
8
7
6
1
2
4
5
3
$\frac{1}{2} d_{6,8}$

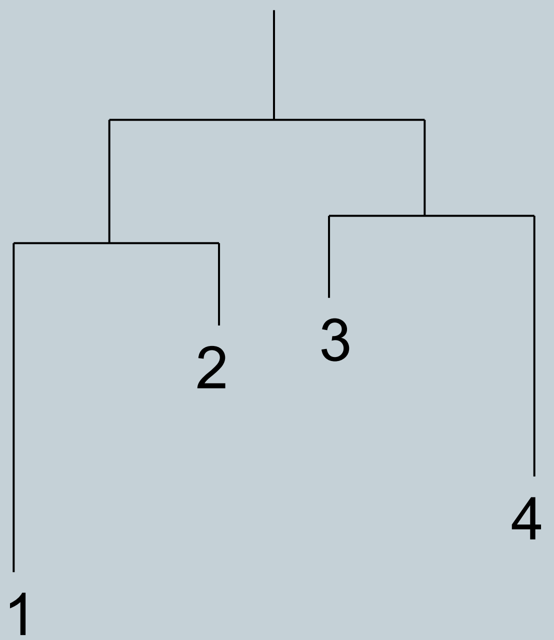
UPGMA implementation

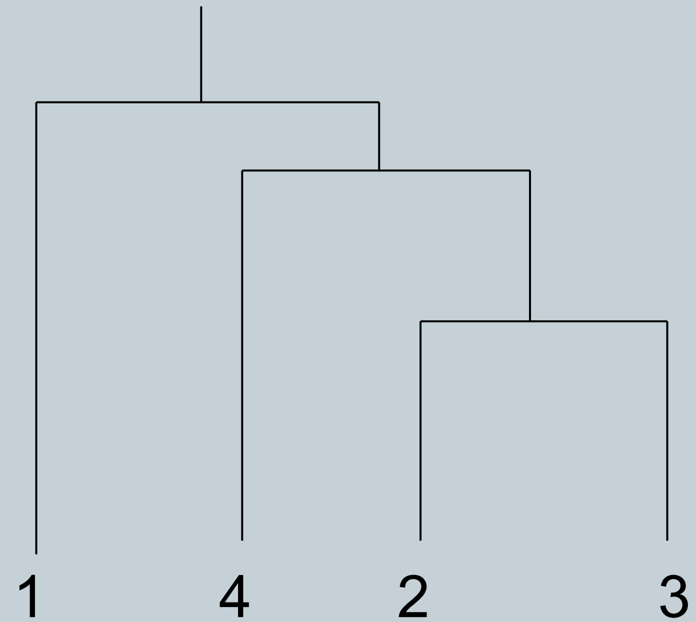
- In naive implementation, each iteration takes $O(n^2)$ time with n sequences => algorithm takes $O(n^3)$ time
- The algorithm can be implemented to take only $O(n^2)$ time (see Gronau & Moran, 2006, for a survey)

Problem solved?

- We now have a simple algorithm which finds a ultrametric tree
  - If the data is ultrametric, then there is exactly one ultrametric tree corresponding to the data (proof left as an exercise)
  - The tree found is then the ”correct” solution to the phylogeny problem, if the assumptions hold
- Unfortunately, the data is not ultrametric in practice
  - Measurement errors distort distances
  - *Basic assumption of a molecular clock does not hold usually very well*

Incorrect reconstruction of non-ultrametric data by UPGMA





Tree which corresponds to non-ultrametric distances

Incorrect ultrametric reconstruction by UPGMA algorithm

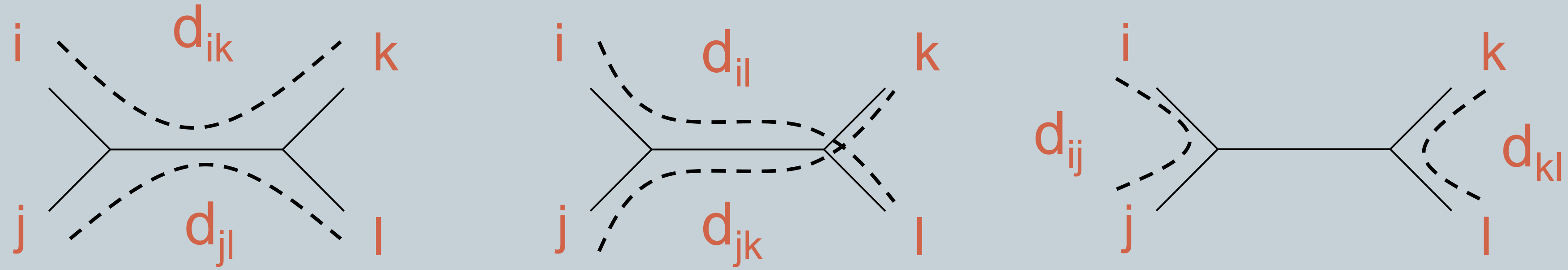
Part II

NEIGHBOR JOINING

Checking for additivity

- How can we check if our data is additive?
- Let i, j, k and l be four *distinct* species
- Compute 3 sums: $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$, $d_{il} + d_{jk}$

Four-point condition



- The sums are represented by the three figures
  - Left and middle sum cover all edges, right sum does not
- *Four-point condition*: i, j, k and l satisfy the four-point condition if two of the sums $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$, $d_{il} + d_{jk}$ are the same, and the third one is smaller than these two

Checking for additivity

- An $n \times n$ matrix D is additive if and only if the four point condition holds for every 4 distinct elements $1 \leq i, j, k, l \leq n$
- See exercises for grounding of three-point (ultrametric) and four-point (additive) conditions.

Finding an additive phylogenetic tree

- Additive trees can be found with, for example, the neighbor joining method (Saitou & Nei, 1987)
- The neighbor joining method produces unrooted trees, which have to be rooted by other means
  - A common way to root the tree is to use an outgroup
  - Outgroup is a species that is known to be more distantly related to every other species than they are to each other
  - Root node candidate: position where the outgroup would join the phylogenetic tree
- However, in real-world data, even additivity usually does not hold very well

Neighbor joining algorithm

- Neighbor joining works in a similar fashion to UPGMA
  - Find clusters C_1 and C_2 that minimise a function $f(C_1, C_2)$
  - Join the two clusters C_1 and C_2 into a new cluster C
  - Add a node to the tree corresponding to C
  - Assign distances to the new branches
- Differences in
  - The choice of function $f(C_1, C_2)$
  - How to assign the distances

Neighbor joining algorithm

- Recall that the distance d_{ij} for clusters C_i and C_j was

$$d_{ij} = \frac{1}{|C_i||C_j|} \sum_{p \in C_i, q \in C_j} d_{pq}$$

- Let $u(C_i)$ be the separation of cluster C_i from other clusters defined by

$$u(C_i) = \frac{1}{n-2} \sum_{C_j} d_{ij}$$

where n is the number of clusters.

Neighbor joining algorithm

- Instead of trying to choose the clusters C_i and C_j closest to each other, neighbor joining at the same time
  - Minimises the distance between clusters C_i and C_j and
  - Maximises the separation of both C_i and C_j from other clusters

Neighbor joining algorithm

- Initialisation as in UPGMA
- Iteration
  - Find clusters i and j for which $d_{ij} - u(C_i) - u(C_j)$ is minimal
  - Define new cluster k by $C_k = C_i \cup C_j$, and define d_{kl} for all l
  - Define a node k with edges to i and j. Remove clusters i and j
  - Assign length $\frac{1}{2} d_{ij} + \frac{1}{2} (u(C_i) - u(C_j))$ to the edge $i \to k$
  - Assign length $\frac{1}{2} d_{ij} + \frac{1}{2} (u(C_j) - u(C_i))$ to the edge $j \to k$
- Termination:
  - When only one cluster remains

Neighbor joining algorithm: example

	a	b	c	d
a	0	6	7	5
b		0	11	9
c			0	6
d				0

i	u(i)
a	(6+7+5)/2 = 9
b	(6+11+9)/2 = 13
c	(7+11+6)/2 = 12
d	(5+9+6)/2 = 10

i,j	$d_{ij} - u(C_i) - u(C_j)$
a,b	6 - 9 - 13 = -16
a,c	7 - 9 - 12 = -14
a,d	5 - 9 - 10 = -14
b,c	11 - 13 - 12 = -14
b,d	9 - 13 - 10 = -14
c,d	6 - 12 - 10 = -16

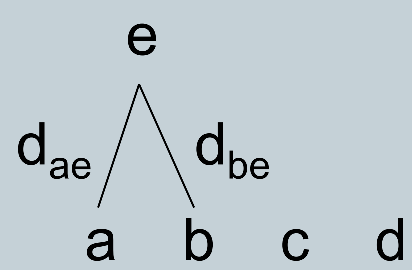
Choose either pair to join

Neighbor joining algorithm: example

	a	b	c	d
a	0	6	7	5
b		0	11	9
c			0	6
d				0

i	u(i)
a	(6+7+5)/2 = 9
b	(6+11+9)/2 = 13
c	(7+11+6)/2 = 12
d	(5+9+6)/2 = 10

i,j	d_{ij} – $u(C_i)$ – $u(C_j)$
a,b	6 – 9 – 13 = -16
a,c	7 – 9 – 12 = -14
a,d	5 – 9 – 10 = -14
b,c	11 – 13 – 12 = -14
b,d	9 – 13 – 10 = -14
c,d	6 – 12 – 10 = -16



d_{ae} = ½ 6 + ½ (9 – 13) = 1

d_{be} = ½ 6 + ½ (13 – 9) = 5

This is the first step only…

Neighbor joining algorithm: correctness

- ***Theorem**: If D is an additive matrix, neighbor joining algorithm correctly constructs the corresponding additive tree.*
  Proof (sketch). By contradiction. Assume i and j with minimum $D_{ij} = d_{ij} - u(C_i) - u(C_j)$ are not neighbors in the additive tree. Show that there are then two neighbors m and n with $D_{mn} < D_{ij}$ (see Durbin et al. *Biological Sequence Analysis*, pp. 190-191 for details). Then the theorem follows by induction.

Study group assignments

WEDNESDAY 12.10. 10-12 B222

CHECK THE FOLLOWING ASSIGNMENTS BEFORE THE STUDY GROUP AND DECIDE WHICH ONES YOU WOULD LIKE TO STUDY THERE:

WWW.CS.HELSINKI.FI/U/VMAKINEN/ALGBIO11/ALGBIO11_STUDYGROUP5.PDF